UNITED STATES DISTRICT COURT
DISTRICT OF MASSACHUSETTS

DATATERN, INC. Plaintiff, v. BLAZENT, INC., MICROSTRATEGY INC., CARL WARREN AND COMPANY INCORPORATED, LANCET SOFTWARE DEVELOPMENT, INC., AIRLINES REPORTING CORP., MAGIC SOFTWARE ENTERPRISES LTD., MAGIC SOFTWARE ENTERPRISES, INC., TERADATA CORPORATION, INFORMATICA CORPORATION, EPICOR SOFTWARE CORPORATION, and PREMIER, INC. Defendants.	1:11-cv-11970-FDS (Consolidated)

MICROSTRATEGY'S MEMORANDUM IN SUPPORT OF ITS MOTION FOR SUMMARY JUDGMENT OF NO INFRINGEMENT

TABLE OF CONTENTS

TABLE OF AUTHORITIES

I. INTRODUCTION

DataTern's infringement case under the patent-in-suit, U.S. Patent No. 6,101,502 (the "'502 patent), has been untenable from the outset. In an effort to run up MicroStrategy's costs and interfere with its customer relationships as settlement leverage, DataTern has delayed confronting the fact that the accused products do not satisfy the "selecting an object model" limitation required by all asserted claims. DataTern should have abandoned its claims nearly a year ago when the Federal Circuit affirmed a claim construction that is dispositive here.

As the Federal Circuit explained, the claimed "object model" must provide a particular object-oriented programming construct called "classes." When forming its infringement allegations, however, DataTern adopted an unreasonably broad interpretation of "object model"—one unmoored from the patent's teachings—that would *not* require classes. Consistent with that interpretation, neither DataTern's original nor amended infringement contentions even purport to allege that MicroStrategy's accused products have an "object model" comprising classes.

DataTern's overbroad reading of its claims—in particular, the "object model" limitation—was never well founded. For more than two years, however, that reading has been baseless. On August 24, 2012, in a case involving the same '502 patent in the Southern District of New York, the court construed the term "object model" to require, *inter alia*, classes. DataTern appealed a judgment of non-infringement based on that construction to the Federal Circuit. On April 1, 2014, the Federal Circuit affirmed the district court's construction because "the patent and all of the record evidence supports the construction of 'object model' to require a set of classes," and subsequently rejected DataTern's petitions for rehearing. DataTern concedes (as it must) that the Federal Circuit's construction for "object model" is binding in this case.

Now that the proper interpretation of the "object model" term is settled, DataTern can no longer maintain its allegations against MicroStrategy's products and a judgment of non-infringement is overdue. DataTern has no infringement theory or evidence to account for the settled "object model" construction. Further, DataTern should be precluded from either claiming a need for additional discovery or seeking leave to amend its infringement contentions with a new theory. DataTern had ample opportunity to take discovery of MicroStrategy's product operation and to present a theory under the foreseeable, correct construction of "object model." DataTern instead put all its eggs in one basket by seeking an overly broad construction. DataTern's attempts to convince the Federal Circuit to adopt that construction failed. For these reasons, as explained further below, summary judgment of non-infringement is required.

II. FACTS

A. Relevant Procedural History

1. Background of DataTern's Massachusetts Litigations and Accusations

Between November 7 and 15, 2011, DataTern filed seventeen lawsuits in this District alleging infringement of the '502 patent. (SOF[1] ¶ 1.) MicroStrategy and many of its customers were named as defendants. Several of those cases have since been dismissed. Today, DataTern's sole remaining actions in this District target MicroStrategy's products and name MicroStrategy or its customers as defendants. (SOF ¶ 2.) The Court has consolidated these remaining cases into this action.

On April 26, 2012, DataTern served MicroStrategy with infringement contentions pursuant to the Local Rules and the Scheduling Order. MicroStrategy informed DataTern that its

[1] "SOF" refers to MicroStrategy's Rule 56.1 Statement of Undisputed Material Facts, filed herewith.

infringement contentions were deficient, among other reasons because they were "unclear as to precisely what aspects of MicroStrategy's software constitutes the … object model." (SOF ¶ 3.). DataTern responded that it "sought production of the source code before providing its infringement contentions" and "suggest[ed] that MicroStrategy provide its source code so that DataTern can supplement its infringement contentions." (SOF ¶ 4.) DataTern also promised that "[c]onsistent with the Joint Statement Pursuant to Local Rules 16.1(D) and 16.6, nonetheless, DataTern intends to supplement its infringement contentions within 90 days of production of MicroStrategy's source code." (SOF ¶ 5.)

In view of DataTern's recalcitrance to spell out its infringement theory, MicroStrategy moved to compel DataTern to provide adequate contentions. (SOF ¶ 6.) DataTern's opposition claimed its lack of access to MicroStrategy's source code prevented it from supplementing, describing the "necessity that review of the source code plays in confirming and defining the allegedly infringing products in software cases." (SOF ¶ 7.) DataTern then purported to reserve its rights to amend its contentions "since it did not have access to the source code." (SOF ¶ 8.)

In its July 2, 2012 Order on MicroStrategy's motion, the Court ordered DataTern to amend its infringement contentions and identify specifically "where each asserted claim element is found within each accused product or combination and supporting evidence." (SOF ¶ 9.) DataTern accordingly served amended contentions on July 20, 2012. (SOF ¶ 10.) MicroStrategy then informed DataTern that its amended infringement contentions remained deficient and failed to demonstrate DataTern's basis for bringing this case. (SOF ¶ 11.) DataTern again replied that it would be "glad to supplement after reviewing the source code when it is produced." (SOF ¶ 12.)

MicroStrategy made source code available to DataTern shortly thereafter, on August 10, 2012. (SOF ¶ 13.) It requested that DataTern—pursuant to the parties' Joint Statement, the Local

Rules, and DataTern's previous pledge—amend its contentions to provide source code citations within ninety days. (SOF ¶ 14.) DataTern responded to MicroStrategy's request stating, for the first time, "we disagree with your contention that DataTern is required to inspect the MicroStrategy source code within 90 days." (SOF ¶ 15.)

Soon thereafter, DataTern moved to stay this case in light of a claim construction order on the same patent issued out of the Southern District of New York. (SOF ¶ 16.) MicroStrategy opposed the stay and requested that DataTern be held to its obligation to provide amended contentions with citations to MicroStrategy's source code. (SOF ¶ 17.) The Court granted the stay, but advised that "[t]he stay shall not, however, prohibit DataTern from inspecting the software source code made available by MicroStrategy." More specifically, at a hearing on DataTern's stay motion in October 2012, the Court warned DataTern:

> "[W]hat I don't want to do is to get to, let's say, January [2013] and then have DataTern say we've got to go inspect the source code and that's going to take 90 days or whatever. In other words, ***if you think you need to [inspect MicroStrategy's source code], the time to do it is now***."

(SOF ¶ 19) (emphasis added). Despite the Court's instruction more than two years ago, DataTern never inspected MicroStrategy's source code and never supplemented its contentions.

2. DataTern's New York Litigations

During the early stages of this litigation, DataTern was also involved in a consolidated lawsuit in the Southern District of New York against Microsoft Corp. and SAP AG involving alleged infringement of DataTern's '502 patent.[2] (SOF ¶ 20.) On August 24, 2012, the New York

[2] DataTern's New York litigations were declaratory judgment actions filed by Microsoft and SAP after DataTern had filed multiple lawsuits in the Eastern District of Texas alleging infringement by numerous Microsoft and SAP customers (similar to its multiple customer lawsuits here). *DataTern, Inc. v. Staples, Inc. et al.*, No. 2:10-cv-133 (E.D. Tex. Apr. 19, 2010); *DataTern, Inc. v. Eli Lilly and Co. et al.*, No. 2:10-cv-413 (E.D. Tex. Oct. 7, 2010); *DataTern, Inc. v. Abbott Labs. et al.*, No. 2:10-cv-203 (E.D. Tex. Apr. 4, 2011).

court construed several terms from the ’502 patent, including “object model.” (SOF ¶ 21.) That court recognized the essential dispute between the parties was whether an “object model” must include classes. (SOF ¶ 22.) The court sided with Microsoft and SAP’s position on this pivotal dispute, determining that an “object model” definitively requires classes, and thus the court construed the “object model” limitation as follows (SOF ¶ 23):

“Object Model” Construction
“a template with a predetermined standardized structure both relating to an object-oriented software application and ***including object classes*** and inheritance relationships among classes.”

DataTern conceded it could not prove infringement in the New York cases under the court’s construction of “object model,” nor under additional construed terms “to create at least one interface object” and “runtime engine.” (SOF ¶ 24.) Judgment entered against DataTern on December 26, 2012 and DataTern appealed to the Federal Circuit. (SOF ¶ 25.)

3. DataTern’s Stipulation of Non-infringement in Massachusetts

Following the adverse judgments in New York, DataTern also conceded it could not prove MicroStrategy’s products infringe the ’502 patent applying the New York court’s claim constructions. (SOF ¶ 26.) Thus, DataTern proposed this Court either issue a judgment of non-infringement, which DataTern would appeal, or stay the case pending DataTern’s appeal of the New York court’s judgment. (SOF ¶ 27.) MicroStrategy agreed that summary judgment of non-infringement was required, but disputed the appropriate bases for the judgment. DataTern proposed the judgment of non-infringement be based ***solely*** on the New York court’s construction of the claim limitation “to create at least one interface object.” (SOF ¶ 28.) In an effort to save costs and avoid multiple trips to the Federal Circuit, MicroStrategy instead sought judgment on several independent bases, since DataTern would be unable to prove infringement under any of the New York court’s claim constructions for the ’502 patent, including that court’s construction of "object model." (SOF ¶ 29.) This Court ultimately adopted DataTern's proposed judgment based solely on the conceded term "to create at least one interface object."[3] (SOF ¶ 30.) DataTern then appealed the judgment.

4. The Outcome of DataTern's Appeals

DataTern's opening briefs in the New York appeals were filed on April 1, 2013. (SOF ¶ 31.) It argued that the New York court erred in its construction of all terms underpinning the judgments of non-infringement, *i.e.*, "object model," "to create at least one interface object," and "runtime engine." (SOF ¶ 32.) DataTern's appeal of the judgment of non-infringement in this case similarly argued that the New York court's claim construction of "to create at least one interface object" was flawed. (SOF ¶ 33.) After the appeal of the instant case was fully briefed, the Federal Circuit stayed the proceedings *sua sponte* pending the outcome of the New York appeals, presumably based on the possibility that they might resolve the overlapping dispute in this case and moot further proceedings. (SOF ¶ 34.)

In the New York appeals, on May 5, 2014[4], the Federal Circuit affirmed the New York court's judgment of non-infringement based on the court's construction of "object model." (SOF ¶ 35.) Specifically, the Federal Circuit agreed with the lower court's determination that an "object model" necessarily requires classes and inheritance relationships between those classes. (SOF ¶ 36.) The Federal Circuit, having affirmed the New York judgment based on the

[3] DataTern should have also conceded noninfringement on the "object model" and "runtime engine" limitations in view of the New York court's construction, but instead it offered no argument and ignored those issues entirely. DataTern's refusal to face all of the issues raised by the New York order belies a piecemeal litigation strategy which is plainly designed to drive up MicroStrategy's litigation costs.

[4] The Federal Circuit's April 1, 2014 Opinion was subsequently withdrawn by the court and replaced with a new opinion on May 5, 2014. The only difference was a dissent that was dropped from the replacement. That dissent related solely to a jurisdictional issue and thus did not concern any issue raised here.

dispositive "object model" construction, did not reach the additional claim construction issues on appeal in the New York case, and thus did not address the single claim limitation underpinning this Court's judgment of non-infringement.

Following the Federal Circuit's affirmance of the New York court's "object model" construction, MicroStrategy wrote DataTern on April 14, 2014 requesting that DataTern drop its allegations given that DataTern had no infringement theory for "object model" that could satisfy the applicable construction. (SOF ¶ 37.) DataTern never responded. On May 1, 2014, MicroStrategy again wrote DataTern pointing out that "DataTern has no colorable infringement theory based on the now-affirmed construction of "object model.'" (SOF ¶ 38.) DataTern again did not respond. Thus, MicroStrategy wrote DataTern a third time on July 28, 2014 and a fourth time on August 11, 2014—both letters after the Federal Circuit denied DataTern's petitions for rehearing—asking DataTern yet again to drop the lawsuit in light of the affirmed "object model" construction. (SOF ¶ 39.) To date, DataTern has refused to disclose how it could maintain infringement allegations under the binding "object model" construction.[5]

Meanwhile, because the Federal Circuit affirmed the New York non-infringement judgments without addressing the limitation that supported the judgment of no infringement in MicroStrategy's case ("to create at least one interface object"), the Federal Circuit lifted the stay of the appeal of this case and heard argument on November 4, 2014. Ultimately, the Federal Circuit reversed the judgment in this case in a non-precedential opinion[6] and remanded for

[5] DataTern's only response to MicroStrategy's multiple demands for it to explain its good-faith basis for continuing suit was sent on August 8, 2014, stating only that DataTern had not conceded non-infringement under the "object model" construction, but providing no explanation for how that limitation could possibly be satisfied. *See* Ex. Q, Aug. 8, 2014 Email fr. Belt to Kessel.

[6] In its opinion, the Federal Circuit found that the New York court's construction of "to create at least one interface object," adopted by this Court by the parties' agreement, did not necessarily

further proceedings. *DataTern, Inc. v. Epicor Software Corp. et al.*, Nos. 2013-1251,-1252, 2014 WL 7234037 (Fed. Cir. 2014).

B. Background of the Relevant Technology

1. Object Oriented Programming and Classes[7]

Object-oriented programming is a software development methodology that organizes an application into a collection of objects, which are defined by "classes." Each object class is written in source code and defines, generically, what kinds of data an object can store ("attributes"), and what tasks ("methods") can be performed by or on the object. For instance, an email application's source code might provide for a Message class and a Contact class. Attributes of a Message class could be, *e.g.*, 'Message ID,' 'Subject,' or 'Sender.' The Message class also defines the data types for each attribute. For instance, a 'Message ID' attribute might be stored as a number and the message 'Subject' might be stored as a string of alphanumeric characters. A class attribute might also be stored according to a type that corresponds to another defined class. For instance, the Message class attribute 'Sender' might be stored according to the 'Contact' class, which has its own defined attributes and methods. An example behavior (or method) for the Message class could be a 'send' method, which would define the necessary operations to send a particular message. The figure below depicts the exemplary Message and Contact Classes; the blue arrow illustrates that the 'Contact' class corresponds to the 'Sender'

require code generation. While MicroStrategy disputes that any accused products satisfy this limitation (or the "runtime engine" limitation) even under the Federal Circuit's opinion, this motion focuses only on the "object model" limitation to simplify resolution, since that construction is now final and plainly not subject to any further appeal.

[7] MicroStrategy also provided a background of the relevant technology in its Feb. 4, 2013 Motion for Summary Judgment of Non-infringement, and provides it again here for the Court's convenience given the lapse of time. MicroStrategy believes this background to be non-controversial and consistent with an expert declaration DataTern submitted in the New York action in support of its claim construction positions. *See* Ex. R Gupta Declaration ¶¶ 9–15.

attribute in the ‘Message’ class:

<u>Message Class</u>

Attributes	Type
Message ID	Number
Subject	String of Characters
Sender	Contact class
Methods	
Message.send ()	

<u>Contact Class</u>

Attributes	Type
First Name	String of Characters
Last Name	String of Characters
Phone	Number
Methods	
Contact.delete()	

“Objects” are working instances of an object class that are created during execution (i.e., at runtime) to hold real data and that can carry out functionality defined in the class methods. For instance, while an email application is running and the user initiates a new email message, the application might instantiate a Message object, which would conform to the template defined by the Message class, and would store values for each attribute, *e.g.,* ‘101,’ ‘Meeting at 10:00 a.m.,’ and ‘Robert Jones” (for attributes Message ID, Subject, and Sender, respectively). Two example Message objects that might be created during execution of an email application appear below.

<u>Message Object #1</u>

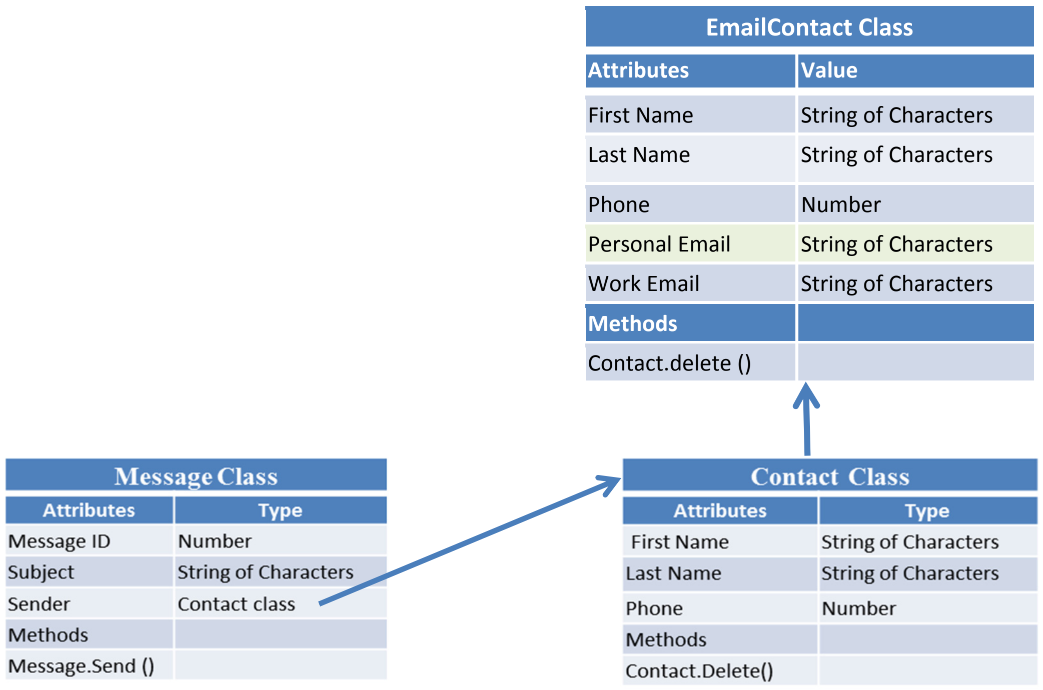
Attributes	Value
Message ID	101
Subject	Meeting at 10:00 a.m.
Sender	Robert Jones
Methods	
Message.send ()	

<u>Message Object #2</u>

Attributes	Value
Message ID	102
Subject	Meeting Cancelled
Sender	Bill Jones
Methods	
Message.send ()	

Since an object-oriented application typically consists of many objects, all defined by the source code classes, an “object model” is used to define the classes that make up an object-oriented application and the relationships between those classes. For instance, in the above example, one such class relationship is that between the ‘Sender’ attribute of the ‘Message’ class and the ‘Contact’ class. Namely, a relationship exists between these classes because each ‘Message’ object will have a ‘Sender’ attribute that points to an object of the ‘Contact’ class.

Another relevant relationship between classes relates to "inheritance." Inheritance creates a "parent/child" relationship between classes wherein the "child" class inherits attributes and methods from the "parent" class, but can also contain additional attributes and behaviors unique to the child class. For instance, the email application used in these examples may also include an 'EmailContact' class that inherits from the Contact class (i.e. the 'Contact' class is the parent class). Thus, the 'EmailContact' class includes all the attributes/behaviors as the 'Contact' class, but also includes additional attributes such as a personal and work email address. An object model associated with the email application would capture each class in the application—along with the class attributes and behaviors—and would represent all relationships between those classes, including inheritance relationships. These example relationships are illustrated below.



EmailContact Class	
Attributes	**Value**
First Name	String of Characters
Last Name	String of Characters
Phone	Number
Personal Email	String of Characters
Work Email	String of Characters
Methods	
Contact.delete ()	

Message Class	
Attributes	**Type**
Message ID	Number
Subject	String of Characters
Sender	Contact class
Methods	
Message.Send ()	

Contact Class	
Attributes	**Type**
First Name	String of Characters
Last Name	String of Characters
Phone	Number
Methods	
Contact.Delete()	

2. Relevant '502 Patent Disclosure

The '502 patent is directed to "interfacing object oriented software applications with relational databases." (SOF ¶ 40.) The patent discloses a specific alleged solution to a perceived need which involves "selecting an object model" and then "mapping between an object model and a relational database." (SOF ¶ 41.) Once the mapping is complete, the object-oriented application can construct objects using information from a relational database (e.g., to provide data values for the object attributes). (SOF ¶ 42.) As the information for an object is modified during application execution, the information can be stored in the relational database, such that the changes are not lost when the application is closed. (SOF ¶ 43.)

The claimed invention begins with, and requires the selection of an "object model" that relates to the object-oriented application. (SOF ¶ 44.) Consistent with the Federal Circuit's claim construction, the patent confirms the patent's "object model" requires classes and inheritance relationships. Figure 3 is the patent's only depiction of the claimed object model (SOF ¶ 45):

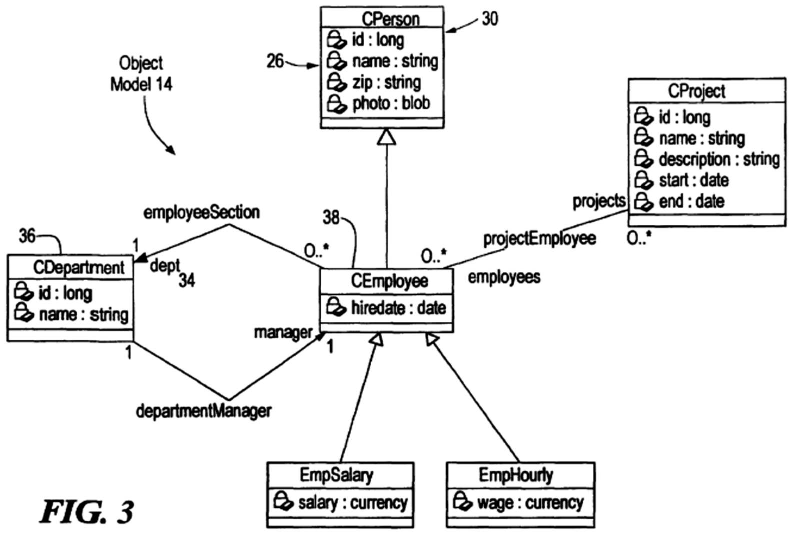


FIG. 3

As the Federal Circuit recognized, Fig. 3's object model "unambiguously" includes classes ("CPerson," "CProject," "CEmployee" and "CDepartment"). (SOF ¶ 46.) Further, the "process described in the '502 patent for mapping the object model to the relational database schema makes clear that the object model must include 'classes' in order to be mapped." (SOF ¶ 47.)

Following selection of the "object model," the "mapping" described in the patent, broadly speaking, means that class attributes within the model are associated with corresponding database table fields. (SOF ¶ 48.) This mapping then permits an object-oriented application (that is associated with the object model) to construct a set of objects based on information stored in a database. (SOF ¶ 49.)

C. DataTern's Allegations against MicroStrategy Products

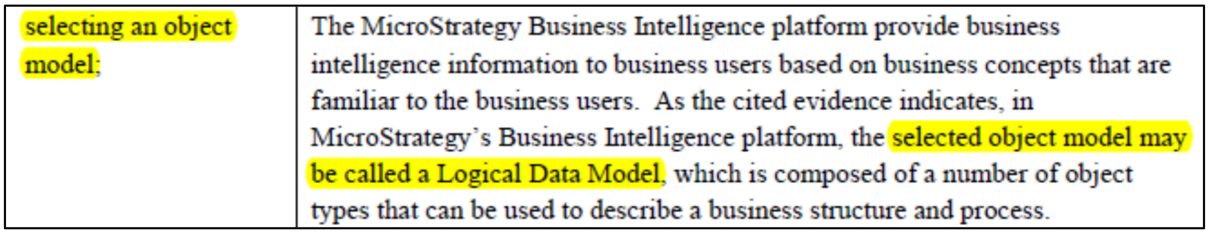
1. MicroStrategy's Business Intelligence Platform

DataTern's amended infringement contentions target MicroStrategy's Business Intelligence Platform. (SOF ¶ 50.) At a high level, that platform facilitates analysis of large volumes of data by providing the ability to view the data from intuitive perspectives, such as in custom summary reports. (SOF ¶ 51.) Before a user can employ the MicroStrategy software platform, he or she needs to think about the types of business data to be analyzed, or included in a report, and how the data elements interrelate. (SOF ¶ 52.) MicroStrategy refers to the output of this conceptualization exercise as a "Logical Data Model." (SOF ¶ 53.) Put another way, the Logical Data Model is a user's understanding of how its business data is arranged, related, and experienced by the general data user or analyst (SOF ¶ 54.) While a user considers the types of data it considers relevant to summary and analysis, and thinks about the relationship between the data, users can categorize data in its Logical Data Model into what MicroStrategy calls Facts and Attributes. (SOF ¶ 55.) "Facts" represent measurements or variables that are typically numerical, such as sales volumes. "Attributes" provide context to Facts, such as the month or country in which a sale took place. (SOF ¶ 56.) Once the user has categorized the types of data the company maintains, the user can employ MicroStrategy's software to capture Facts and Attributes and ultimately use them as building blocks for a report or analysis. (SOF ¶ 57.)

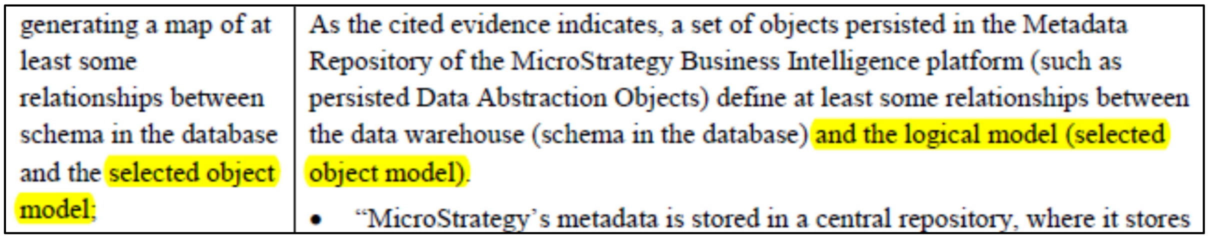
2. DataTern's Infringement Contentions

As explained above, DataTern has served two sets of infringement contentions.

DataTern's amended contentions—like its original contentions—allege that the Logical Data Model satisfies the "object model" limitation, which is required by all claims (SOF ¶ 58):



selecting an object model;	The MicroStrategy Business Intelligence platform provide business intelligence information to business users based on business concepts that are familiar to the business users. As the cited evidence indicates, in MicroStrategy's Business Intelligence platform, the selected object model may be called a Logical Data Model, which is composed of a number of object types that can be used to describe a business structure and process.



generating a map of at least some relationships between schema in the database and the selected object model;	As the cited evidence indicates, a set of objects persisted in the Metadata Repository of the MicroStrategy Business Intelligence platform (such as persisted Data Abstraction Objects) define at least some relationships between the data warehouse (schema in the database) and the logical model (selected object model). • "MicroStrategy's metadata is stored in a central repository, where it stores

The evidence DataTern relies on in its contentions confirms that the Logical Data Model is simply a user's understanding of how various components of that user's business data relate to one another. For example, DataTern's contentions point to MicroStrategy literature explaining the "logical data model is a logical arrangement of data ***as experienced by the general user or business analyst***" (SOF ¶ 59) (emphasis added). Additional documents DataTern cites in support of its allegation that the Logical Data Model is the claimed "object model" describe that "[a] logical data model represents the definition, characteristics, and relationships of data in a technical, conceptual, or business environment. This process can help you think about the various elements that compose your company's business data and how those elements relate to one another." (SOF ¶ 60.)

Classes, a required element of the "object model" per the Federal Circuit's opinion, are an object-oriented programming concept that provide attributes and methods. The accused Logical Data Model, by contrast, is an abstraction reflecting a user's arbitrary organization of its business data. Notably, the Logical Data Model does ***not*** provide software "classes" or "inheritance relationships among classes." Nor does DataTern even purport to assert that it does. This single fact is dispositive of DataTern's infringement claims.

III. ARGUMENT

A. Legal Standards

Summary judgment is appropriate where "the movant shows that there is no genuine dispute as to any material fact and the movant is entitled to a judgment as a matter of law." Fed. R. Civ. P. 56(a). Because DataTern has the burden to prove infringement, MicroStrategy can carry its burden on summary judgment by demonstrating a lack of evidence to support DataTern's allegations. *Regents of Univ. of Cal. v. Dako N. Am., Inc.*, 2009 WL 1083446, at *5 (N.D. Cal. Apr. 22, 2009) ("On an issue for which the opposing party will have the burden of proof at trial, the moving party need only point out that there is an absence of evidence to support the nonmoving party's case.") (internal quotations omitted); *Golden Bridge Technology, Inc. v. Apple Inc.*, 758 F.3d 1362, 1367 (Fed. Cir. 2014) ("We affirm summary judgment of noninfringement where there is an absence of evidence to support the patentee's case.") (citing *Exigent Tech. Inc. v. Atrana Solutions Inc.*, 442 F.3d 1301, 1307–10 (Fed. Cir. 2006).)

Although a non-movant may oppose a summary judgment motion by stating that "it cannot present facts essential to justify its opposition," Fed. R. Civ. P. 56(d), a submission under this rule must "demonstrate that movant has been diligent in conducting discovery, and show 'good cause' why the additional discovery was not previously practicable with reasonable diligence." *Simas v. First Citizens' Fed. Credit Union*, 170 F.3d 37, 45 n.4 (1st Cir. 1999).

B. DataTern Cannot Show that MicroStrategy's Accused Products Select an "Object Model"

All asserted claims require an "object model." Namely, sole independent claims 1 and 10 require "selecting an object model" and a "selected object model" respectively. It is now settled that the claimed "object model" must include object classes. Despite ample opportunity, DataTern has adduced no evidence nor offered any theory to show that the accused products select an "object model" that includes classes. *Golden Bridge Technology, Inc. v. Apple Inc.*, 758 F.3d 1362, 1367 (Fed. Cir. 2014) ("We affirm summary judgment of noninfringement where there is an absence of evidence to support the patentee's case.").

To date, DataTern has never alleged that MicroStrategy's products include an "object model" that includes object classes. Instead, DataTern's amended contentions allege that the "Logical Data Model" referred to in MicroStrategy's product literature is the claimed "object model." (*See supra*, § II.C.2.) As the evidence cited in DataTern's contentions demonstrates, the Logical Data Model is merely an interface for a user to specify how various components of that user's business data relate to one another. *Id.* MicroStrategy's Logical Data Model indisputably lacks object classes (as well as inheritance relationships). And DataTern has never advanced an alternative infringement theory under which the "object model" limitation could be satisfied.

DataTern has provided no evidence to show infringement by the accused products, thus summary judgment is warranted. *Homeland Housewares, LLC v. Sorensen Research*, 581 Fed. Appx. 869, 874 (Fed. Cir. 2014) (finding "[defendant] was not required to produce affirmative evidence of noninfringement. Instead, as the moving party that would not have the burden of proof at trial, [defendant] needed only to point out to the district court the absence of evidence to support the nonmoving party's case.") (internal quotations omitted).

C. DataTern Should Not Be Permitted to Amend Its Infringement Contentions with a New Theory or Request Discovery to Attempt to Find One

Given that DataTern has not provided any infringement theory or evidence under which MicroStrategy's products could satisfy the "object model" limitation, DataTern will likely seek leave to conduct further discovery or again redo its infringement contentions. The Court should reject any such request because DataTern was well aware that "object model" would likely be construed to require object classes and yet DataTern manifested a lack of diligence by failing to account for the likely construction in its infringement contentions and discovery efforts.

Generally speaking, "[t]he purpose of infringement contentions is to provide notice of the plaintiff's theories of infringement early in the case." *Sloan Valve Company v. Zurn Industries, Inc.*, No. 10–cv–00204, 2014 WL 51293 at *2 (N.D. Ill. 2014). Courts restrict the free amendment of infringement contentions "to require parties to crystallize their theories of the case early in the litigation and to adhere to those theories once they have been disclosed." *Acer, Inc. v. Tech. Props. Ltd.*, No. 08–cv–05398, 2010 WL 3618687, at *4 (N.D. Cal. Sept.10, 2010); *see also Convolve, Inc. v. Compaq Computer Corp*., No. 00-civ-5141, 2006 WL 2527773 at *4 (S.D.N.Y. 2006) (stating "the philosophy underlying amendment of invalidity or infringement contentions is 'decidedly conservative' and designed to ensure that litigants do not continue to shift their theories after each claim construction ruling"). In some instances, amendments to infringement contentions are permitted after the court issues an unexpected claim construction. *Nike, Inc. v. Adidas Am., Inc.*, 479 F.Supp.2d 664, 667–68 (E.D.Tex.2007). However, "[a] party cannot argue that because its precise proposal for a construction of a claim term is not adopted by the court, it is surprised and must prepare new infringement contentions." *Id.*

Further, where a party is aware of a foreseeable claim construction, the party should be diligent in accounting for that construction in its infringement contentions or be prepared to accept the consequences of an adverse construction. *See St. Clair Intellectual Property Consultants, Inc. v. Matsushita Elec. Indus. Co., Ltd*., No. 04–1436, 2012 WL 1015993 at *5 (D. Del. 2012). In *St. Clair*, the plaintiff served infringement contentions that did not account for defendants' proposed construction. *Id.* at *6. Plaintiff did so despite the fact that (1) the USPTO has adopted defendants' proposed construction in a prior reexamination (2) Plaintiff's preferred construction, although adopted in a separate litigation, was being reviewed by the Federal Circuit, and (3) claim construction in the co-pending litigation had not occurred when Plaintiff served its infringement contentions. *Id.* Ultimately, the Federal Circuit construed the term adversely to the plaintiff, and the defendants moved for summary judgment. Plaintiff proposed that it be allowed to devise new infringement theories in light of the adverse construction. *Id.* at *5. The court denied the plaintiff's request and granted summary judgment stating:

> [t]he Court concludes that St. Clair has not demonstrated diligence in its assertion of its infringement contentions. Specifically, at the time that St. Clair served its infringement contentions and expert reports, claim construction in this case had not yet been resolved. When claim construction remains an open issue at the time the parties serve expert reports and infringement contentions, the parties have an obligation to prepare for the fact that the court may adopt the other party's claim construction.

Id. Like the *St. Clair* patentee, DataTern had ample notice that "object model" would likely be construed to require object classes. Back in 2008, DataTern's predecessor-in-interest to the '502 patent, FireStar Software, in fact ***proposed*** a construction of "object model" that required classes. *See Microsoft Corp. v. DataTern, Inc.*, 755 F.3d 899, 909 (Fed. Cir. 2014). In December 2011, in a case pending in the Eastern District of Texas, the defendants accused by DataTern of infringing the '502 patent proposed a similar construction for "object model" that also required classes. Ex. S, Joint Claim Construction Statement at Ex. A, 3-4. In DataTern's New York cases, in April 2012, the accused infringers again proposed a construction for "object model" that required classes. Ex. T, Joint Claim Construction Statement; Ex. U, Scheduling Order. Although there has not been a separate *Markman* proceeding in the instant case, MicroStrategy made clear to DataTern, at least as early as May 2012, that MicroStrategy understood "object model" to include classes. ECF No. 43-1 at 4-5. In fact, an "object model" construction that requires classes was not only foreseeable, it was inevitable. The New York court found "there is no way to read in a possibility that an object model is not connected to or representing classes; it is mandatory, not discretionary" and "to fulfill the central purpose of the patent, the object model must include classes." Ex. G, New York Claim Construction Order, at 13. Similarly, the Federal Circuit found that "the '502 patent makes clear that the object model must include classes in order to practice the claimed invention," "the patent and all of the record evidence supports the construction of 'object model' to require a set of classes," and "the only embodiment in the patent discloses an object model with classes." *Microsoft Corp. v. DataTern, Inc.*, 755 F.3d at 909.

Despite DataTern's knowledge that "object model" would likely be construed to require classes, DataTern failed to account for the likely construction in its infringement contentions—no doubt because there is no way to read the claims on the accused products under that construction. Neither DataTern's original nor its amended infringement contentions account for the correct construction of "object model" as neither purports to identify an "object model" comprised of classes and inheritance relationships. Notably, DataTern's amended contentions were served on July 20, 2012, long after DataTern received proposed constructions in the New York case (where the accused infringers proposed that an "object model" required classes), and after MicroStrategy also disclosed to DataTern that its interpretation of "object model" required classes. Even following the New York court's "object model" construction—at which point MicroStrategy wrote DataTern requesting that DataTern supplement its contentions to account for the New York constructions—DataTern elected not to supplement its contentions to account for the ordered construction. Ex. V, Sept. 12, 2012 Letter from Brown to Zucker.

DataTern should also not be permitted to prolong final resolution of this case by demanding discovery or source code inspection. First and foremost, no additional discovery or inspection would save DataTern's case because MicroStrategy's "Logical Data Model" plainly lacks anything resembling the claimed "object model." Indeed, if DataTern believed MicroStrategy's products allowed selection of an "object model" comprised of classes, surely DataTern would have (a) pursued discovery to support such a theory, *e.g.* by timely inspecting MicroStrategy's code when it was made available, and (b) disclosed that theory in its infringement contentions. DataTern did neither. Instead, as soon as it became apparent that "object model" would be construed to require classes, DataTern abandoned its earlier promise to review MicroStrategy's source code and supplement its contentions.

As explained above (*supra* § II.A.1), early in the case DataTern committed to reviewing MicroStrategy's source code. DataTern first agreed to a case schedule under which MicroStrategy would produce source code and DataTern would supplement its contentions with source code citations within ninety days. DataTern also repeatedly sought to justify the deficiencies in its infringement contentions based on the "necessity that review of the source code plays in confirming and defining the allegedly infringing products in software cases." *Id.* Only after the New York court's construction of "object model" did DataTern completely reverse course and for the first time take the position that it was not required to review MicroStrategy's source code and provide corresponding citations.

MicroStrategy raised this issue with the Court both in motion practice and during the October 5, 2012 hearing, requesting that the Court confirm that DataTern was required to supplement its infringement contentions with source code citations. The Court declined to specifically order inspection, but advised DataTern that ***if*** DataTern believed it needed to inspect source code, "the time to do it is now." *Id.* Bound by the New York court's construction and this Court's warning, DataTern still never inspected the source code. The time for reviewing source code has passed. DataTern's own lack of diligence has left it without evidence of infringement and thus DataTern must answer for its tactical decision. *Convolve, Inc. v. Compaq Computer Corp.*, No. 00-civ-5141, 2006 WL 2527773 at *4 (S.D.N.Y. 2006) (disallowing amendments to a party's contentions because "parties who are on notice of the opposing party's claim construction and do not prepare for the fact that the court may adopt it can be held accountable").

In the end, instead of exercising diligence to develop an infringement theory under the correct claim construction (an exercise which would have proved fruitless in any event), DataTern gambled that it could reverse the New York construction on appeal. DataTern made its litigation tactic plain on multiple occasions as it told MicroStrategy and this Court that over 40% of claim constructions are reversed at the Federal Circuit. D.I. 83 at 12; No. 11-11970, D.I. 30 (citing the "substantial doubt that the NY claim construction will be upheld if for no other reason than that forty percent of those parties aggrieved by Markman decisions by district courts, who seek appellate review, obtain relief"); Ex. W, Apr. 4, 2013 Letter from Zucker to Brown (stating the reversal statistics at the Federal Circuit "run somewhere between 40 and 50 percent"). The Federal Circuit, however, affirmed the "object model" construction, which now binds the parties here. And DataTern has no viable infringement claim against MicroStrategy, making summary judgment appropriate.

IV. CONCLUSION

For the reasons set forth above, MicroStrategy requests this Court grant summary judgment of non-infringement in favor of Defendants.

Dated: February 11, 2015

<u>/s/ Adam J. Kessel</u>
Adam Kessel (BBO #661211)
FISH & RICHARDSON P.C.
One Marina Park Drive
Boston, MA 02110-1878
Tel: (617) 542-5070
Fax: (617) 542-8906
kessel@fr.com

Benjamin K. Thompson (admitted *pro hac vice*)
FISH & RICHARDSON P.C.
1180 Peachtree St. NE
Atlanta, GA 30318
Tel: (404) 724-2844
bthompson@fr.com

Attorneys for Defendant
MICROSTRATEGY INCORPORATED.

<u>CERTIFICATE OF SERVICE</u>

I hereby certify that this document(s) filed through the ECF system will be sent electronically to the registered participants as identified on the Notice of Electronic Filing (NEF) and paper copies will be sent to those indicated as non-registered participants on this 11th of February, 2015.

<u>/s/ Adam J. Kessel</u>
Adam J. Kessel